IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of: Debjit Das Sarma	§	Group Art Unit: 2193
	§	
	§	Examiner: Ngo, Chuong D.
Serial No. 10/730,800	§	
	§	Atty. Dkt. No.: 5500-97400
Filed: December 9, 2003	§	
	§	****CERTIFICATE OF E-FILING TRANSMISSION**** I hereby certify that this correspondence is being transmitted via electronic filing to the United States Patent and Trademark Office on the date shown below.
	§	Anthony M. Petro (Printed Name)
For: Apparatus and Method for Multiple Pass Extended Precision Floating Point Multiplication	§	/Anthony M. Petro/ (Signature) August 18, 2008 (Date)

APPEAL BRIEF

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir/Madam:

Further to the Notice of Appeal filed June 17, 2008, Appellant presents this Appeal Brief. Appellant respectfully requests that the Board of Patent Appeals and Interferences consider this appeal.

I. REAL PARTY IN INTEREST

As evidenced by the assignment recorded at Reel/Frame 014772/0940, the subject application is owned by Advanced Micro Devices, Inc., a corporation organized and existing under and by virtue of the laws of the State of Delaware, and having its principal place of business at One AMD Place, Sunnyvale, CA, 94088.

II. RELATED APPEALS AND INTERFERENCES

No other appeals, interferences or judicial proceedings are known which would be related to, directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

Claims 23-28 were previously canceled. Claims 1-22 are pending in the application and stand finally rejected. The rejection of claims 1-22 is being appealed. A copy of claims 1-22 is included in the Claims Appendix hereinbelow.

IV. STATUS OF AMENDMENTS

An amendment canceling claims 26-28 was submitted subsequent to the Final Rejection. In an Advisory Action mailed July 15, 2008, the Examiner indicated that the amendment would be entered for purposes of appeal. No amendments to the specification or drawings have been submitted subsequent to the Final Rejection.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Independent claim 1 is directed to a floating point multiplier circuit configured for performing extended-precision multiplication of an N-bit multiplicand value by an M-bit multiplier value, wherein N and M are positive integers (*see, e.g.*, FIG. 2, multiplier 200; and p. 7, lines 2-16, p. 8, lines 24-26).

The floating point multiplier circuit includes partial product generation logic configured to generate a plurality of partial products from the multiplicand value and the multiplier value (*see, e.g.*, FIG. 2, partial product generation logic 220; and p. 8, lines 17 – p. 9, line 4). The plurality of partial products corresponds to a first portion of the multiplier value during a first partial product execution phase, and wherein the plurality of partial products further corresponds to a second portion of the multiplier value during a second partial product execution phase (*see, e.g.*, FIG. 3, FIG. 5, blocks 502-504; and p. 10, lines 5-22).

The floating point multiplier circuit also includes a plurality of carry save adders coupled to the partial product generation logic (*see, e.g.*, FIG. 2, carry save adder logic 230; and p. 11, line 6 – p. 12, line 4). The carry save adders are configured to accumulate the plurality of partial products generated during the first partial product execution phase into a redundant product during a first carry save adder execution phase, and further configured to accumulate the plurality of partial products generated during the second partial product execution phase into the redundant sum during a second carry save adder execution phase (*see, e.g.*, FIG. 3, FIG. 5, blocks 506-508; and p. 13, lines 6-21).

The floating point multiplier circuit further includes a first carry propagate adder coupled to the plurality of carry save adders (*see, e.g.*, FIG. 2, carry propagate adder 240; and p. 14, lines 14-21). The first carry propagate adder is configured to reduce a first portion of the redundant product to a multiplicative product during a first carry propagate adder phase, and further configured to reduce a second portion of the redundant product to the multiplicative product during a second carry propagate adder phase (*see, e.g.*, FIG.

3, FIG. 5, blocks 510-512; and p. 14, lines 21-26). The first carry propagate adder phase begins after the second carry save adder execution phase completes (*see, e.g.*, FIG. 3; and p. 15, lines 17-26).

Independent claim 10 is directed to a method of operation of a multiplier circuit (*see, e.g.*, FIG. 2, multiplier 200, FIG. 5) that includes the multiplier circuit receiving an N-bit multiplicand value and an M-bit multiplier value (*see, e.g.*, FIG. 5, block 500; and p. 24, lines 1-6).

The method also includes the multiplier circuit generating a plurality of partial products from the multiplicand value and the multiplier value (*see, e.g.*, FIG. 2, partial product generation logic 220; and p. 8, lines 17 – p. 9, line 4), wherein the plurality of partial products corresponds to a first portion of the multiplier value during a first partial product execution phase, and wherein the plurality of partial products further corresponds to a second portion of the multiplier value during a second partial product execution phase (*see, e.g.*, FIG. 3, FIG. 5, blocks 502-504; and p. 10, lines 5-22, p. 24, lines 6-10).

The method further includes the multiplier circuit accumulating the plurality of partial products generated during the first partial product execution phase into a redundant product during a first carry save adder execution phase, and the multiplier circuit accumulating the plurality of partial products generated during the second partial product execution phase into the redundant product during a second carry save adder execution phase (*see, e.g.*, FIG. 3, FIG. 5, blocks 506-508; and p. 13, lines 6-21, p. 24, lines 12-19).

The method further includes the multiplier circuit reducing a first portion of the redundant product to a multiplicative product during a first carry propagate adder phase, and the multiplier circuit reducing a second portion of the redundant product to the multiplicative product during a second carry propagate adder phase (*see, e.g.*, FIG. 3, FIG. 5, blocks 510-512; and p. 14, lines 21-26, p. 24, line 21 – p. 21, line 2). The first

carry propagate adder phase begins after the second carry save adder execution phase completes (*see, e.g.*, FIG. 3; and p. 15, lines 17-26).

Independent claim 16 is directed to a microprocessor comprising dispatch logic configured to issue multiply instructions to a floating-point unit, and a floating-point unit coupled to the dispatch logic (*see, e.g.*, FIG. 1, microprocessor 100, dispatch logic 120, floating point unit 140; and p. 6, lines 3-28).

The floating point unit is configured to receive an N-bit multiplicand value and an M-bit multiplier value (*see, e.g.*, FIG. 5, block 500; and p. 24, lines 1-6) and to generate a plurality of partial products from the multiplicand value and the multiplier value (*see, e.g.*, FIG. 2, partial product generation logic 220; and p. 8, lines 17 – p. 9, line 4). The plurality of partial products corresponds to a first portion of the multiplier value during a first partial product execution phase, and wherein the plurality of partial products further corresponds to a second portion of the multiplier value during a second partial product execution phase (*see, e.g.*, FIG. 3, FIG. 5, blocks 502-504; and p. 10, lines 5-22).

The floating point unit is further configured to accumulate (*see, e.g.*, FIG. 2, carry save adder logic 230; and p. 11, line 6 – p. 12, line 4) the plurality of partial products generated during the first partial product execution phase into a redundant product during a first carry save adder execution phase, and to accumulate the plurality of partial products generated during the second partial product execution phase into the redundant product during a second carry save adder execution phase (*see, e.g.*, FIG. 3, FIG. 5, blocks 506-508; and p. 13, lines 6-21).

The floating point unit is further configured to reduce (*see, e.g.*, FIG. 2, carry propagate adder 240; and p. 14, lines 14-21) a first portion of the redundant product to a multiplicative product during a first carry propagate adder phase, and to reduce a second portion of the redundant product to the multiplicative product during a second carry propagate adder phase (*see, e.g.*, FIG. 3, FIG. 5, blocks 510-512; and p. 14, lines 21-26).

The first carry propagate adder phase begins after the second carry save adder execution phase completes (*see, e.g.*, FIG. 3; and p. 15, lines 17-26).

The summary above describes various examples and embodiments of the claimed subject matter. However, the claims are not necessarily limited to any of these examples and embodiments. The claims should be interpreted based on their respective wording.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Claims 1-22 stand rejected under 35 U.S.C. § 101 as allegedly being directed to nonstatutory subject matter.

VII. ARGUMENT

First ground of rejection:

The Examiner rejected claims 1-22 under 35 U.S.C. § 101, alleging that the claims are directed to non-statutory subject matter. Final Action at 3-4. Specifically, the Examiner asserts that the claims "merely involve calculations and manipulations of data," that the result produced by the claims lacks "a real world value" that is "useful, concrete and tangible." *Id.* at 4. Further, the Examiner alleges that the claims are "directed to a preemption of the claimed manipulation and calculation of data." *Id.* Appellant respectfully traverses this rejection for at least the following reasons. Different groups of claims are argued under their respective subheadings.

Claims 1-9:

1. Unlike *Benson*, the claimed subject matter is tied to a specifically recited apparatus.

In rejecting claim 1, the Examiner offers a passing citation to *Gottschalk v. Benson*, 409 U.S. 64 (1972). Final Action at 3. However, Appellant notes that claim 1 is distinguishable from the claims at issue in *Benson*. The claims in *Benson* were directed towards a method for converting BCD numerals into pure binary numerals. *Benson*, 409 U.S. at 254. The Supreme Court specifically noted that Benson's claims "were not limited to any particular art or technology, to any particular apparatus or machinery, or to any particular end use." *Id.* Rather, "[t]hey purported to cover any use of the claimed method in a general-purpose digital computer of any type." *Id.* It is precisely because Benson's claims were not limited to any particular technology, apparatus, or machinery that the Supreme Court was able to hold that, if patented, the subject matter of Benson's method claims would "in practical effect . . . be a patent on the algorithm itself." *Id.* at 257.

Unlike in *Benson*, claim 1 is not directed towards a method in the abstract. Rather, claim 1 is directed towards an apparatus, and more specifically, a floating point multiplier circuit. Further, the subject matter of claim 1 is in fact limited to specifically recited apparatus. For example, claim 1 recites that elements of the floating point multiplier circuit include "partial product generation logic," "a plurality of carry save adders," and "a first carry propagate adder." Moreover, these elements are coupled to each other in specifically recited ways.

Thus, claims 1 is explicitly limited to particular apparatus. In contrast, the claims at issue in *Benson* were not so limited. Therefore, *Benson* is factually distinguishable from the present case.

2. The claimed subject matter is consistent with the requirements of *Alappat*, *State Street*, and *AT&T*.

In *State Street*, which is perhaps the most well-known recent case to address statutory subject matter issues, the Federal Circuit observed that "[u]npatentable mathematical algorithms are identifiable by showing that they are merely abstract ideas constituting disembodied concepts or truths that are not 'useful.' . . . [T]his means that to be patentable an algorithm must be applied in a 'useful' way." *State Street Bank & Trust Co. v. Signature Financial*, 149 F.3d 1368, 1373 (Fed. Cir. 1998). In this instance, it is helpful to consider *State Street* in the context of its companion cases. In particular, the Federal Circuit relied upon *In re Alappat*, which the *State Street* panel characterized as follows: "In *Alappat*, we held that data, transformed by a machine through a series of mathematical calculations to produce a smooth waveform display on a rasterizer monitor, constituted a practical application of an abstract [algorithm], because it produced 'a useful, concrete[,] and tangible result'—the smooth waveform." *Id.* (citing *In re Alappat*, 33 F.3d 1526, 1544 (Fed. Cir. 1996)).

The "useful, concrete, and tangible" language of *Alappat* is often quoted without considering the context of its two preceding paragraphs. Given the similarity between the

claims at issue in *Alappat* and the present case, this omitted context is particularly instructive when attempting to understand the rationale underlying the test.

The claims in *Alappat* involved a "particularly claimed combination of elements performing the particularly claimed combination of calculations to transform . . . digitized waveforms into anti-aliased, pixel illumination data to produce a smooth waveform." *Alappat*, 33 F.3d at 1544. The panel began its analysis as to whether Alappat's claims fell within the ambit of the mathematical subject matter exception to 35 U.S.C. § 101 by stating that "the proper inquiry . . . is to see whether the claimed subject matter *as a whole* is a disembodied mathematical concept." *Id.* (emphasis in original).

The *Alappat* panel concluded this inquiry in the negative:

> Although many, or arguably even all, of the means elements recited in claim 15 represent circuitry elements that perform mathematical calculations, which is essentially true of all digital electrical circuits, the claimed invention as a whole is directed to a combination of interrelated elements which combine to form a machine for converting discrete waveform data samples into anti-aliased pixel illumination intensity data to be displayed on a display means. This is not a disembodied mathematical concept which may be characterized as an "abstract idea," but rather a specific machine to produce a useful, concrete, and tangible result.

Id. (emphasis added). That is, the *Alappat* panel concluded that a collection of circuit elements that combine to form a machine that produces a particular kind of mathematical result (i.e., the pixel illumination intensity data) is statutory subject matter. *Alappat* specifically notes that while the claims themselves implement various mathematical concepts, they are not directed to the disembodied mathematical concepts that are applied, but are instead directed to the specific machine that implements them.

It is critical to note that the rationale of *Alappat* is not comprehensively reflected by its quoted portion in *State Street*. That is, the *State Street* panel's characterization of *Alappat* as finding statutory subject matter because the claims "produced 'a useful,

concrete[,] and tangible result'" suggests that this feature was the sole determinant of the conclusion. However, this is not the case. As noted above, the holding of *Alappat* relies as much on the fact that the claims are directed to a specific machine as on the nature of the result produced by the machine. While it is true, as the *State Street* panel notes, that the nature of the result is a consideration with respect to statutory subject matter, it is incorrect to conclude that under *Alappat*, the nature of the result is the sole consideration. To distinguish the claims at issue from mere "disembodied mathematical concept[s]," the *Alappat* panel relied as much on the fact that the claims were directed to "a specific machine" as it did on the fact that this machine "produce[s] a useful, concrete, and tangible result." *Alappat*, 33 F.3d at 1544. Nothing in *Alappat* suggests that either factor was considered more significant, important, or necessary than the other.

In a later case, the Federal Circuit expounded upon the logic and application of the *Alappat* inquiry, first clarifying that "the judicially-defined proscription against patenting of a 'mathematical algorithm,' to the extent such a proscription still exists, is narrowly limited to mathematical algorithms in the abstract." *AT&T Corp. v. Excel Communications, Inc.*, 172 F.3d 1352, 1356 (Fed. Cir. 1999) (emphasis added). The panel then noted that "the *Alappat* inquiry simply requires an examination of the contested claims to see if the claimed subject matter as a whole is a disembodied mathematical concept representing nothing more than . . . an 'abstract idea,' or if the mathematical concept has been reduced to some practical application rendering it 'useful.'" *Id.* at 1357. The *AT&T* panel concluded that since AT&T's claimed process "applies a Boolean principle to produce a useful, concrete, tangible result without pre-empting other uses of the mathematical principle, on its face the claimed process comfortably falls within the scope of § 101." *Id.* at 1358.

In understanding the logic of *AT&T*, it is essential to note that the claims at issue in that case were method claims that recited no explicit implementing structure. *Id.* at 1354-55. That is, the claims at issue recited only method actions such as "generating a record" and "including . . . [an] indicator [in said record]." *Id.* Had the *AT&T* claims been directed to a particular machine that recited implementing structure, the result

would have followed directly from *Alappat* with little fanfare. However, lacking this option, the *AT&T* panel nonetheless decided that the recited application was specific enough, even in the absence of implementing structure, to be statutory. That is, finding no "specific machine" in the claims, the *AT&T* panel concluded that the application recited in the claims was sufficient. It cannot be concluded from the holding of *AT&T* that this factor was necessary even in the presence of implementing structure, since these were not the facts of *AT&T*. That is, *AT&T* cannot be read to narrow the holding of *Alappat* to requiring only a useful, concrete, tangible result as both a necessary and sufficient condition for statutory subject matter, regardless of the presence of implementing structure, since the claims in *AT&T* lacked any mention of implementing structure.

The holding of *State Street* is not to the contrary. The *State Street* panel observed that "the mere fact that a claimed invention involves inputting numbers, calculating numbers, outputting numbers, and storing numbers, in and of itself, would not render it nonstatutory subject matter, unless, of course, its operation does not produce a 'useful, concrete and tangible result.'" *State Street*, 149 F.3d at 1374. However, this proviso refers only to a hypothetical claim directed to the process of inputting, calculating, outputting, and storing numbers. Like the claims at issue in *AT&T*, the hypothetical drawn by the *State Street* panel lacks any reference to implementing structure, but instead refers to operations on numbers however implemented. Since such a claim constitutes nothing more than bare process steps, attention necessarily turns to the result produced by the claim, since there is nothing more of substance that is addressed by the claim. But this does not entail that the presence of a useful, concrete, and tangible result is both a necessary and sufficient condition for statutory subject matter even in the presence of implementing structure, since the scenario addressed by the *State Street* panel in their quoted aside does not include such structure.

The *Alappat* rationale, as viewed through the lenses of *AT&T* and *State Street*, is clear: claims that are directed to "disembodied mathematical concepts"—that is, an abstract expression or formula—do not fall within the ambit of § 101. This is, however, a

narrow exception that ensnares only claims that, as a whole, represent nothing more than an abstract expression or formula. *Alappat* provides that a claim directed to a specific machine that produces a useful, concrete, and tangible result falls outside the exception. *AT&T* and *State Street* narrow the exception still further by providing that, even in the absence of implementing structure, bare method actions that produce a useful, concrete, tangible result are nonetheless statutory.

Although *Benson* is factually distinguishable from claim 1 as noted previously, the above analysis is nevertheless consistent with the facts in *Benson*. The claims at issue in *Benson* recited bare method actions without implementing structure. *Benson*, 409 U.S. at 258-59. Although Benson's claim 8 recited a "reentrant shift register," the shift register was recited as a passive storage element that served as a repository for the data operated upon by the claim, rather than as an active structure that carried out the actions recited in the claim. *Id.* In every other respect, the claims in *Benson* were directed to mathematical operations performed in the abstract, without reference to any specific machine that performs those operations. *Benson* thus falls outside the scope of *Alappat*, which requires some mention of a specific machine, and is more closely aligned with the facts in *AT&T*. Unlike *AT&T*, however, the result in *Benson* is indistinguishable from an ordinary algebraic variable such as y in the expression $y = a + b$. The result of the claims in *Benson* is not explicitly tied to a particular application. Further, there is no result implicit in the claims that arises from the manner in which the actions are performed. That is, there is no useful property or quality, such as improved performance or decreased cost, that arises from the recited configuration of actions in *Benson*. Thus, the claims of *Benson* are indeed little more than "disembodied mathematical concepts," representative of nothing more than an abstract expression or formula, and as such fall outside the scope of § 101 according to *Alappat* and its successors.

Having assessed the relevant case law, it remains to apply the principles of these precedents to the instant claims. Claim 1 is directed to a floating point multiplier circuit configured for performing extended-precision multiplication of an N-bit multiplicand value by an M-bit multiplier value, where N and M are positive integers. The multiplier

circuit includes partial product generation logic configured to generate a plurality of partial products from the multiplicand value and the multiplier value in a particularly recited manner over two distinct phases of operation of the partial product generation logic. The multiplier circuit further includes a plurality of carry save adders coupled to the partial product generation logic and configured to accumulate the plurality of partial products generated during the two distinct phases of operation of the partial product generation logic into a redundant product over two distinct phases of operation of the carry save adders. The multiplier circuit further includes a first carry propagate adder coupled to the plurality of carry save adders and configured to reduce first and second portions of the redundant product to a multiplicative product during first and second carry propagate adder phases. Finally, claim 1 further constrains the operation of the recited circuit by requiring the first carry propagate adder phase to begin after the second carry save adder execution phase completes.

Claim 1 falls squarely within the holding of *Alappat*, in that it is directed to a specific machine that is configured to produce a useful, concrete, and tangible result. The claims in *Alappat* included a number of means elements corresponding to arithmetic logic elements configured to produce the recited illumination intensity data as a result. *Alappat*, 33 F.3d at 1538-39. Similarly, the instant claims recite a "particularly claimed combination of elements performing the particularly claimed combination of calculations" to produce a result. *Id.* at 1544. In the case of claim 1, these elements include specific types of adder circuits (carry save adders and a carry propagate adder) as well as partial product generation logic. There can be little doubt that, as in *Alappat*, the recitations of claim 1 constitute a specific machine for producing its result. Unlike *Benson*, claim 1 does not merely recite disembodied actions that are not implemented by any particular kind of structure. To the contrary, claim 1 contains explicit mention of particular structural features interrelated and limited in specifically recited ways in order to implement their recited functions.

Moreover, the result of claim 1 is useful, concrete, and tangible. First, this claim is not directed to the operation of multiplication in the abstract, such as may be expressed

by an equation such as $y = ab$. It is more specifically and concretely recited than a mere formula, in that it does not merely describe how the result is produced in terms of abstract operations, but instead describes the characteristics of a particular machine that is configured to produce the result. Moreover, the result of claim 1 is not simply that a product of two binary numbers is produced. Rather, it is that the product is produced by circuitry that is reused during the course of producing the product. That is, during a first phase of operation, partial product generation logic produces partial products corresponding to a first portion of a multiplier value, while during a second phase of operation, the same partial product generation logic produces partial products corresponding to a different portion of the multiplier value. Similarly, subsequent adder logic operates during multiple phases of operation to accumulate and reduce the partial products to a final multiplicative product. Thus, the result of the claimed structure is not merely that a multiplicative product is produced in the abstract. Rather, the result of the claim, considering it as a whole as required by *Alappat*, is that such a product is produced in a machine in which recited structures are reused.

As explained in Appellant's specification, providing hardware support for extended-precision floating-point multiplication within a microprocessor is costly in terms of die size and power consumption. Specification at para. [0005]-[0006]. Reuse of hardware resources for multiplication as recited in the claims may therefore ameliorate such area and power costs relative to a different hardware implementation. This effect is also a result of the claimed configuration of hardware. Further, such reduced costs undeniably have real-world implications, since they directly affect the manufacturing economics of an integrated circuit that includes the recited circuitry. A design that is more efficient and therefore reduces manufacturing costs unquestionably produces useful, concrete, and tangible results, because manufacturing costs are real, meaningful, and measurable factors of integrated circuit production rather than simply abstract algebraic variables.

Claim 1 recites a further specific limitation: that the first phase of operation of the carry propagate adder begins after the second phase of operation of the carry save adder

completes. This has the necessary consequence that the carry propagate adder logic and the carry save adder logic are not concurrently active during the processing of a given multiplication operation. Because the operation of these two types of adders does not overlap for a given multiplication, it necessarily follows that under certain circumstances, the peak power consumption of the multiplier circuit may be lower than in an embodiment where these two types of adders operated concurrently for a given multiplication. Dynamic power consumption (i.e., the rate at which circuit power consumption changes over time) is a significant factor in integrated circuit design, in that rapid changes in power consumption may create rapid changes in electrical current. In turn, these changes in current may create unwanted electrical or electromagnetic effects (such as voltage droop, resistive heating, noise, and/or radiated emissions) that may negatively affect circuit operation or may require additional design effort and/or manufacturing resources to accommodate. Therefore, a reduction of the effects of dynamic power consumption in at least some circumstances presents another useful, concrete, and tangible result of the claimed circuit arrangement.

As noted in *AT&T*, the proscription against claims directed to "disembodied mathematical concepts" is a narrow one that applies only to those claims that represent nothing more than an abstract expression or formula. The instant claims stand for substantially more than such disembodied concepts. As demonstrated above, they recite particular structures that form a specific machine for producing one or more useful, concrete, and tangible results. As such, claim 1 satisfies the criteria for statutory subject matter articulated in *Alappat* and clarified in *State Street* and *AT&T*.

.

3. The claimed subject matter does not preempt an underlying algorithm.

The proscription against claims directed to nothing more than abstract expressions or formulas has an adjunct requirement: an apparatus claim must not be "'so abstract and sweeping' [in its implementation of an underlying algorithm] that it would 'wholly pre-empt' the use of any apparatus employing the combination of mathematical calculations recited therein." *Alappat*, 33 F.3d at 1544 (quoting *Benson*, 409 U.S. at 68-72). The

Alappat panel found no such preemption, owing to the fact that the claim at issue was "limited to the use of a particularly claimed combination of elements performing the particularly claimed combination of calculations." *Id.* Similarly, in *AT&T*, the Federal Circuit noted that the claims at issue did rely upon a "simple [Boolean] mathematical principle" to derive a result. *AT&T*, 172 F.3d at 1358. However, the panel also noted that this fact alone "was not determinative because AT&T does not claim the Boolean principle as such or attempt to forestall its use in any other application." *Id.*

Precisely the same situation as in *Alappat* applies to claim 1. That is, claim 1 is limited to a particularly claimed combination of elements: the recited types of adders and partial product generation logic. Further, these elements perform a particularly claimed combination of calculations to generate and accumulate partial products into a result. This result is determined under particularly recited constraints or conditions (e.g., the reuse of underlying hardware and the timing constraints mentioned above) that contribute effects on the cost of producing the result.

Similarly, as in *AT&T*, claim 1 makes use of underlying mathematical principles such as accumulating partial products to produce a multiplicative product. However, also as in *AT&T*, claim 1 is not directed to these principles as such. Nor does it prevent the use of these principles in any other application. For example, it is possible that a general purpose computer could be programmed to implement each of the functional aspects recited in claim 1 via software instructions in a manner that is entirely independent of the hardware configuration recited in the claims. That is, such a computer could use ordinary arithmetic and logical instructions to implement the generation of partial products and their accumulation to form a multiplicative product. Such an implementation would not infringe the instant claims, since it would not implement floating-point multiplication via the recited structures, but rather via software. Because another embodiment exists that could implement the principles employed in the instant claims, yet would fall outside the scope of the instant claims, then by definition the instant claims cannot be said to preempt all uses of the underlying algorithm.

Thus, for at least the foregoing reasons, Appellant submits that claim 1 is directed to a specific machine that produces a useful, concrete, tangible result and does not preempt the use of any apparatus implementing the underlying algorithm. As such, Appellant submits that claim 1 as well as its dependent claims are directed to statutory subject matter as required by 35 U.S.C. § 101.

Claims 10-15:

Claim 10 differs from claim 1 in that claim 10 is a method claim directed to a method of operation of a multiplier circuit. The method of claim 10 recites each of the functional attributes of the multiplier circuit of claim 1, phrased as actions that are performed by the explicitly-recited multiplier circuit.

Benson is inapplicable to the method of claim 10. As noted above with respect to claim 1, the method claims in *Benson* were not limited to any particular sort of apparatus. By contrast, claim 10 explicitly requires that the various recited actions are performed by a multiplier circuit. According to the relevant definition in the Merriam-Webster Online Dictionary, a "circuit" is "an assemblage of electronic elements." Electronic elements are necessarily some type of concrete apparatus. Therefore, a "multiplier circuit" is an assemblage of concrete electronic elements that is specifically adapted to perform multiplication. A purely mechanical calculator configured to perform multiplication would not be a multiplier circuit. Therefore, claim 10 is not directed to an abstract sequence of steps that is not tied to a particular machine, but is instead limited to a particularly recited apparatus.

The rationale of *Alappat* is applicable to claim 10 despite the fact that claim 10 is directed to a method. Unlike the claims in *AT&T*, which were method claims that recited no implementing structure, claim 10 recites a specific machine—the multiplier circuit—that performs the recited actions. As noted above, *Alappat* provides that claims that are directed to "disembodied mathematical concepts"—that is, claims that as a whole, stand for nothing more than an abstract expression or formula—do not fall within the ambit of

§ 101, while a claim directed to a specific machine that produces a useful, concrete, and tangible result is statutory.

Claim 10 is directed to such a specific machine performing the recited actions. Further, claim 10 recites a specific limitation that produces a useful, concrete, and tangible result: it requires that the first phase of operation of the carry propagate adder begins after the second phase of operation of the carry save adder completes. As noted above with respect to claim 1, this has the necessary consequence that the carry propagate adder logic and the carry save adder logic are not concurrently active during the processing of a given multiplication operation. Because the operation of these two types of adders does not overlap for a given multiplication, it necessarily follows that under certain circumstances, the peak power consumption of the multiplier circuit may be lower than in an embodiment where these two types of adders operated concurrently for a given multiplication. Dynamic power consumption (i.e., the rate at which circuit power consumption changes over time) is a significant factor in integrated circuit design, in that rapid changes in power consumption may create rapid changes in electrical current. In turn, these changes in current may create unwanted electrical or electromagnetic effects (such as voltage droop, resistive heating, noise, and/or radiated emissions) that may negatively affect circuit operation or may require additional design effort and/or manufacturing resources to accommodate. Therefore, a reduction of the effects of dynamic power consumption in at least some circumstances presents another useful, concrete, and tangible result of the claimed method of operation of the recited multiplier circuit.

Moreover, claim 10 does not preempt all use of the underlying algorithm. Claim 10 is not directed to the underlying mathematical principles as such, but instead is specifically limited to an electronic multiplier circuit. Multipliers implemented in other, non-electronic technologies would not fall within to scope of claim 10. Moreover, a general purpose computer that does not include a multiplier circuit could implement any or all of the functional aspects recited in claim 10 via purely software instructions. As noted above with respect to claim 1, such a computer could use ordinary arithmetic and

logical instructions to implement the generation of partial products and their accumulation to form a multiplicative product. Such an implementation would not infringe claim 10, since the actions performed by the general-purpose computer would not be performed by a multiplier circuit, but instead by different types of circuits (e.g., an adder circuit and/or a Boolean logic circuit). Because other embodiments exist that could implement the principles employed in the instant claims, yet would fall outside the scope of the instant claims, then by definition the instant claims cannot be said to preempt all uses of the underlying algorithm.

Thus, for at least the foregoing reasons, Appellant submits that claim 10 is directed to a method of operation of a specific machine that produces a useful, concrete, tangible result and does not preempt the use of any apparatus implementing the underlying algorithm. As such, Appellant submits that claim 10 as well as its dependent claims are directed to statutory subject matter as required by 35 U.S.C. § 101.

Claims 16-22:

Claim 16 is similar to claim 1 in that both claims are directed towards specifically recited configurations of apparatus elements. Claim 16 is directed to a microprocessor comprising specific hardware elements: dispatch logic configured to issue multiply instructions to a floating-point unit; and a floating-point unit coupled to the dispatch logic. The recited floating-point unit is configured to implement the same functional elements as the multiplier circuit of claim 1.

As argued above with respect to claim 1, *Benson* is inapplicable to the microprocessor of claim 16. In contrast to *Benson*, which was not limited to any particular sort of apparatus, claim 16 explicitly requires that the recited functional elements are implemented by a floating-point unit that is included within a microprocessor and is coupled to dispatch logic that is configured to issue multiply instructions to the floating-point unit. Unlike *Benson*, claim 16 recites a specifically-recited configuration of particular hardware elements.

Claim 16, like claim 1, falls squarely within the holding of *Alappat*, in that it is directed to a specific machine that is configured to produce a useful, concrete, and tangible result. Like the claims in *Alappat*, claim 16 recites a "particularly claimed combination of elements performing the particularly claimed combination of calculations" to produce a result. *Alappat*, 33 F.3d at 1544. That is, claim 16 recites dispatch logic coupled to, and configured to issue multiply instruction to, a floating-point unit. Thus, as in *Alappat*, the recitations of claim 16 constitute a specific machine for producing its result. Unlike *Benson*, claim 16 does not merely recite disembodied actions that are not implemented by any particular kind of structure. To the contrary, claim 16 contains explicit mention of particular structural features interrelated and limited in specifically recited ways in order to implement their recited functions.

Moreover, the result of claim 16 is useful, concrete, and tangible. First, this claim is not directed to the operation of multiplication in the abstract, such as may be expressed by an equation such as $y = ab$. It is more specifically and concretely recited than a mere formula, in that it does not merely describe how the result is produced in terms of abstract operations, but instead describes the characteristics of a particular machine that is configured to produce the result. Additionally, claim 16 recites a specific limitation that produces a useful, concrete, and tangible result: it requires that the first phase of operation of the carry propagate adder begins after the second phase of operation of the carry save adder completes. As noted above with respect to claim 1, this has the necessary consequence that the carry propagate adder logic and the carry save adder logic are not concurrently active during the processing of a given multiplication operation. Because the operation of these two types of adders does not overlap for a given multiplication, it necessarily follows that under certain circumstances, the peak power consumption of the multiplier circuit may be lower than in an embodiment where these two types of adders operated concurrently for a given multiplication. Dynamic power consumption (i.e., the rate at which circuit power consumption changes over time) is a significant factor in integrated circuit design, in that rapid changes in power consumption may create rapid changes in electrical current. In turn, these changes in current may

create unwanted electrical or electromagnetic effects (such as voltage droop, resistive heating, noise, and/or radiated emissions) that may negatively affect circuit operation or may require additional design effort and/or manufacturing resources to accommodate. Therefore, a reduction of the effects of dynamic power consumption in at least some circumstances presents another useful, concrete, and tangible result of the claimed method of operation of the recited multiplier circuit.

Claim 16 does not preempt an underlying algorithm. As with claim 1, claim 16 is limited to a particularly claimed combination of elements: the recited combination of dispatch logic and a floating-point unit. Further, these elements perform a particularly claimed combination of calculations to generate and accumulate partial products into a result. This result is determined under particularly recited constraints or conditions (e.g., the timing constraints mentioned above) that contribute effects on the cost of producing the result.

While claim 16 makes use of mathematical principles, claim 16 is not directed to these principles as such. Nor does it prevent the use of these principles in any other application. For example, microprocessors that lack specific hardware support for floating-point arithmetic in the form of a floating-point unit are well known in the art. It is further well known that such integer-only microprocessors can be programmed to emulate floating-point arithmetic via code routines that use only integer instructions. Such a microprocessor could be programmed to implement each of the functional aspects recited in claim 16 via software instructions in a manner that is entirely independent of the hardware configuration recited in the claims. That is, such a computer could use ordinary arithmetic and logical instructions to implement the generation of partial products and their accumulation to form a multiplicative product. Such an implementation would not infringe the instant claims, since it would not implement floating-point multiplication via a floating-point unit coupled to dispatch logic, but rather in an emulated fashion via software. Because another embodiment exists that could implement the principles employed in the instant claims, yet would fall outside the scope of the instant claims, then by definition the instant claims cannot be said to preempt all

uses of the underlying algorithm.

Thus, for at least the foregoing reasons, Appellant submits that claim 16 is directed to a specific machine that produces a useful, concrete, tangible result and does not preempt the use of any apparatus implementing the underlying algorithm. As such, Appellant submits that claim 16 as well as its dependent claims are directed to statutory subject matter as required by 35 U.S.C. § 101.

CONCLUSION

For the foregoing reasons, it is submitted that the Examiner's rejection of claims 1-22 was erroneous, and reversal of this decision is respectfully requested.

The Commissioner is authorized to charge the appeal brief fee of $510.00 and any other fees that may be due to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5500-97400/AMP.

Respectfully submitted,

/Anthony M. Petro/
Anthony M. Petro, Reg. #59,391
Agent for Appellant

Meyertons, Hood, Kivlin, Kowert & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
(512) 853-8800

Date: August 18, 2008

VIII. CLAIMS APPENDIX

The claims on appeal are as follows.

1. A floating point multiplier circuit configured for performing extended-precision multiplication of an N-bit multiplicand value by an M-bit multiplier value, wherein N and M are positive integers, said floating point multiplier circuit comprising:

partial product generation logic configured to generate a plurality of partial products from said multiplicand value and said multiplier value, wherein said plurality of partial products corresponds to a first portion of said multiplier value during a first partial product execution phase, and wherein said plurality of partial products further corresponds to a second portion of said multiplier value during a second partial product execution phase;

a plurality of carry save adders coupled to said partial product generation logic and configured to accumulate said plurality of partial products generated during said first partial product execution phase into a redundant product during a first carry save adder execution phase, and further configured to accumulate said plurality of partial products generated during said second partial product execution phase into said redundant sum during a second carry save adder execution phase; and

a first carry propagate adder coupled to said plurality of carry save adders and configured to reduce a first portion of said redundant product to a multiplicative product during a first carry propagate adder phase, and further configured to reduce a second portion of said redundant product to said multiplicative product during a second carry propagate adder phase;

wherein said first carry propagate adder phase begins after said second carry save adder execution phase completes.

2. The floating point multiplier circuit as recited in claim 1, wherein:

said plurality of carry save adders is further configured to perform an arithmetic left shift on said redundant product accumulated during said first carry save adder execution phase by a number of bits corresponding to said first portion of said multiplier value; and

said plurality of carry save adders is further configured to accumulate a result of said arithmetic left shift with said second portion of said plurality of partial products into said redundant product during said second carry save adder execution phase.

3. The floating point multiplier circuit as recited in claim 1, wherein:

said first portion of said multiplier value corresponds to a higher-order portion of said multiplier value;

said second portion of said multiplier value corresponds to a lower-order portion of said multiplier value;

said first portion of said redundant product corresponds to a lower-order portion of said redundant product; and

said second portion of said redundant product corresponds to a higher-order portion of said redundant product.

4. The floating point multiplier circuit as recited in claim 1, wherein:

said redundant product includes a Q-bit sum term and an R-bit carry term;

said first carry propagate adder includes a plurality of operand inputs, wherein each operand input includes at most P bits; and

each of P, Q, and R is a positive integer, P is less than Q, and P is less than R.

5. The floating point multiplier circuit as recited in claim 1 further comprising a plurality of rounding adders coupled to said plurality of carry save adders and configured to produce a respective plurality of rounded multiplicative products.

6. The floating point multiplier circuit as recited in claim 5, wherein each rounding adder is further configured to:

receive a respective rounding constant;

accumulate said respective rounding constant with a first portion of said redundant product into a rounded redundant product during said first carry propagate adder phase;

reduce a first portion of said rounded redundant product to a given respective rounded multiplicative product during said first carry propagate adder phase; and

reduce a second portion of said rounded redundant product to said given respective rounded multiplicative product during said second carry propagate adder phase.

7. The floating point multiplier circuit as recited in claim 1, further configured for performing pipelined reduced-precision multiplication of said N-bit multiplicand value by an S-bit multiplier value with a single partial product execution phase, a single carry save adder execution phase, and a single carry propagate adder phase, wherein S is a positive integer and S is less than or equal to N/2, and wherein each

of said single partial product execution phase, said single carry save adder execution phase, and said single propagate adder phase is operable to accept a new reduced-precision multiplication operation during a given execution cycle.

8. The floating point multiplier circuit as recited in claim 1, wherein said partial product generation logic includes a plurality of Booth encoders and a plurality of Booth multiplexers.

9. The floating point multiplier circuit as recited in claim 1, wherein M is equal to 2Y, wherein said first portion of said multiplier value includes the most significant Y bits of said multiplier value, and wherein said second portion of said multiplier value includes the least significant Y bits of said multiplier value.

10. A method of operation of a multiplier circuit, comprising:

said multiplier circuit receiving an N-bit multiplicand value and an M-bit multiplier value;

said multiplier circuit generating a plurality of partial products from said multiplicand value and said multiplier value, wherein said plurality of partial products corresponds to a first portion of said multiplier value during a first partial product execution phase, and wherein said plurality of partial products further corresponds to a second portion of said multiplier value during a second partial product execution phase;

said multiplier circuit accumulating said plurality of partial products generated during said first partial product execution phase into a redundant product during a first carry save adder execution phase;

said multiplier circuit accumulating said plurality of partial products generated during said second partial product execution phase into said redundant product during a second carry save adder execution phase;

said multiplier circuit reducing a first portion of said redundant product to a multiplicative product during a first carry propagate adder phase; and

said multiplier circuit reducing a second portion of said redundant product to said multiplicative product during a second carry propagate adder phase;

wherein said first carry propagate adder phase begins after said second carry save adder execution phase completes.

11. The method as recited in claim 10, further comprising :

said multiplier circuit performing an arithmetic left shift on said redundant product accumulated during said first carry save adder execution phase by a number of bits corresponding to said first portion of said multiplier value; and

said multiplier circuit accumulating a result of said arithmetic left shift with said second portion of said plurality of partial products into said redundant product during said second carry save adder execution phase.

12. The method as recited in claim 10, wherein:

said first portion of said multiplier value corresponds to a higher-order portion of said multiplier value;

said second portion of said multiplier value corresponds to a lower-order portion of said multiplier value;

said first portion of said redundant product corresponds to a lower-order portion of said redundant product; and

said second portion of said redundant product corresponds to a higher-order portion of said redundant product.

13. The method as recited in claim 10, wherein:

said redundant product includes a Q-bit sum term and an R-bit carry term;

each of said first and second portion of said redundant product includes at most P bits; and

each of P, Q, and R is a positive integer, P is less than Q, and P is less than R.

14. The method as recited in claim 10, further comprising:

said multiplier circuit receiving a plurality of rounding constants;

said multiplier circuit accumulating each rounding constant with a first portion of said redundant product into a respective rounded redundant product during said first carry propagate adder phase;

said multiplier circuit reducing a first portion of each said respective rounded redundant product to a given respective rounded multiplicative product during said first carry propagate adder phase; and

said multiplier circuit reducing a second portion of each said respective rounded redundant product to said given respective rounded multiplicative product during said second carry propagate adder phase.

15. The method as recited in claim 10, further comprising said multiplier circuit selectively performing pipelined reduced-precision multiplication of said N-bit multiplicand value by an S-bit multiplier value with a single partial product execution phase, a single carry save adder execution phase, and a single carry propagate adder phase, wherein S is a positive integer and S is less than or equal to N/2, and wherein each of said single partial product execution phase, said single carry save adder execution phase, and said single propagate adder phase is operable to accept a new reduced-precision multiplication operation during a given execution cycle.

16. A microprocessor comprising:

dispatch logic configured to issue multiply instructions to a floating-point unit; and

a floating-point unit coupled to said dispatch logic and configured to:

receive an N-bit multiplicand value and an M-bit multiplier value;

generate a plurality of partial products from said multiplicand value and said multiplier value, wherein said plurality of partial products corresponds to a first portion of said multiplier value during a first partial product execution phase, and wherein said plurality of partial products further corresponds to a second portion of said multiplier value during a second partial product execution phase;

accumulate said plurality of partial products generated during said first partial product execution phase into a redundant product during a first carry save adder execution phase;

accumulate said plurality of partial products generated during said second partial product execution phase into said redundant product during a second carry save adder execution phase;

reduce a first portion of said redundant product to a multiplicative product during a first carry propagate adder phase; and

reduce a second portion of said redundant product to said multiplicative product during a second carry propagate adder phase;

wherein said first carry propagate adder phase begins after said second carry save adder execution phase completes.

17. The microprocessor as recited in claim 16, wherein:

said plurality of carry save adders is further configured to perform an arithmetic left shift on said redundant product accumulated during said first carry save adder execution phase by a number of bits corresponding to said first portion of said multiplier value; and

said plurality of carry save adders is further configured to accumulate a result of said arithmetic left shift with said second portion of said plurality of partial products into said redundant product during said second carry save adder execution phase.

18. The microprocessor as recited in claim 16, wherein:

said first portion of said multiplier value corresponds to a higher-order portion of said multiplier value;

said second portion of said multiplier value corresponds to a lower-order portion of said multiplier value;

said first portion of said redundant product corresponds to a lower-order portion of said redundant product; and

said second portion of said redundant product corresponds to a higher-order portion of said redundant product.

19. The microprocessor as recited in claim 16, wherein:

said redundant product includes a Q-bit sum term and an R-bit carry term;

said first carry propagate adder includes a plurality of operand inputs, wherein each operand input includes at most P bits; and

each of P, Q, and R is a positive integer, P is less than Q, and P is less than R.

20. The microprocessor as recited in claim 16 further comprising a plurality of rounding adders coupled to said plurality of carry save adders and configured to produce a respective plurality of rounded multiplicative products.

21. The microprocessor as recited in claim 20, wherein each rounding adder is further configured to:

receive a respective rounding constant;

accumulate said respective rounding constant with a first portion of said redundant product into a rounded redundant product during said first carry propagate adder phase;

reduce a first portion of said rounded redundant product to a given respective rounded multiplicative product during said first carry propagate adder phase; and

reduce a second portion of said rounded redundant product to said given respective rounded multiplicative product during said second carry propagate adder phase.

22. The microprocessor as recited in claim 16, further configured for performing pipelined reduced-precision multiplication of said N-bit multiplicand value by an S-bit multiplier value with a single partial product execution phase, a single carry save adder execution phase, and a single carry propagate adder phase, wherein S is a positive integer and S is less than or equal to N/2, and wherein each of said single partial product execution phase, said single carry save adder execution phase, and said single propagate adder phase is operable to accept a new reduced-precision multiplication operation during a given execution cycle.

IX. EVIDENCE APPENDIX

No evidence submitted under 37 CFR §§ 1.130, 1.131 or 1.132 or otherwise entered by the Examiner is relied upon in this appeal.

X. RELATED PROCEEDINGS APPENDIX

There are no related proceedings.